

Orbits of linear maps and regular languages

S. Tarasov*
serge99meister@gmail.com

M. Vyalyi†
vyalyi@gmail.com

December 5, 2010

We settle the equivalence between the problem of hitting a polyhedral set by the orbit of a linear map and the intersection of a regular language and a language of permutations of binary words ($P_{\mathbb{B}}$ -realizability problem).

The decidability of the both problems is presently unknown and the first one is a straightforward generalization of the famous Skolem problem and the nonnegativity problem in the theory of linear recurrent sequences.

To show a ‘borderline’ status of $P_{\mathbb{B}}$ -realizability problem with respect to computability we present some decidable and undecidable problems closely related to it.

This paper is an extended version of the journal publication [16] and contains some additional results.

Introduction

Let Φ be a linear map of a vector space V into itself and let $x \in V$ be a vector in V . The iterations of Φ applied to x define an *orbit* $\text{Orb}_{\Phi} x$, i.e. the set

$$\{\Phi^k x : k \in \mathbb{Z}^+\}.$$

In the present paper we discuss algorithmic issues related to orbits. We assume that V is a rational coordinate space, so Φ and x are represented by their (rational) components.

An *orbit description problem* consists in finding specific relations which either hold for all vectors in the orbit or are violated by at least one vector in the orbit. Here we limit ourselves to a simple case where the relations are formed from Boolean combinations of linear equalities and inequalities (see the exact definition below in Section 2).

Note that the most important case of the orbit problem is *the chamber hitting problem*. In this particular case we check whether an orbit intersects a closed polyhedron (i.e. a solution set for a finite system of nonstrict linear inequalities).

*Supported in part by RFBR grant 08–01–00414.

†Supported by RFBR grant 09–01–00709 and the scientific school grant NSH5294.2008.1.

The orbit description problems are related to some problems on linear recurrent sequences.

A *linear recurrent sequence* (LRS) x_n of a degree d is defined by:

$$\begin{cases} x_n = \sum_{i=1}^d a_i x_{n-i} & \text{when } n > d, \\ x_n = b_n & \text{when } 1 \leq n \leq d, \end{cases} \quad \text{where } a_i, b_j \text{ are constants.} \quad (1)$$

The famous Skolem problem is perhaps the most known algorithmic problem on linear recurrences.

The Skolem problem. Let $\{x_n\}$ be a specified LRS with integer coefficients. Whether $x_k = 0$ for some k ?

The decidability of the Skolem problem is presently an open question, but it is known that it is decidable for the degrees ≤ 5 (the cases $d = 3, 4$ are worked out by N. Vereshchagin [17], and the case $d = 5$ is solved by V. Halava et al. [7]).

In the opposite direction, the best hardness result on the Skolem problem is NP-hardness [3].

LRS is called *nonnegative* if all its elements are nonnegative. One more important algorithmic question about LRS is called the *nonnegativity problem* and consists in checking nonnegativity of a LRS. This problem is at least as hard as the Skolem problem. Being a bit more formal it means that the Skolem problem is Turing reducible to the nonnegativity problem. This statement certainly belongs to the public mind, but see the proof of the statement in Section 4.

The nonnegativity problem is decidable for LRS of the degree ≤ 3 (V. Laohakosol, P. Tangsupphathawat [9]).

In this paper we often use some standard constructions from the theory of algorithms and the complexity theory, e.g., Turing reducibility, m -reducibility, polynomial reducibility. In particular, Turing reducibility of a *Problem I* to *Problem II* means that while solving *Problem I* the reduction may ask an oracle, who can give answers to *Problem II* (see, further [13, 6].)

The relations between the orbit description problems and LRS are described in Section 2.

These results are certainly not new but we include them for completeness sake and in order to introduce necessary notation and terminology.

Let briefly sketch the contents of the paper.

We recall basic properties of LRS in Section 1. LRS are closely related to regular languages. In particular, any LRS can be represented as a *difference* of the generating functions of a pair of regular languages (see, e.g., [11, Cor. 8.2]). We will use a modification of this result (see Theorem 3 from Section 4). (All necessary facts about regular languages could be found in [8, 2].)

The main result of the present paper consists in an algorithmic equivalence between the chamber hitting problem and checking a particular property of the regular languages. Namely, the property involved consists in checking whether a regular language contains at least one word from a special set of words. We call this set a *permutation filter*. Speaking informally, an arbitrary word from the permutation filter gives a

permutation of all binary words of a fixed length n . See a formal statement in Section 3 below.

The permutation filter is denoted by $P_{\mathbb{B}}$ and the corresponding problem of checking this property of regular languages is called $P_{\mathbb{B}}$ -realizability problem (see, Section 3).

The proof of an algorithmic equivalence of a problem of $P_{\mathbb{B}}$ -realizability and the chamber hitting problem (Theorem 2 from Section 3) proceeds in several steps.

At first, we describe in Section 4 a polynomial reducibility of the chamber hitting problem to the $P_{\mathbb{B}}$ -realizability problem. The reducibility uses the aforementioned Theorem 3.

As a consequence of this result we prove (see, theorem 4) NP-hardness of a $P_{\mathbb{B}}$ -realizability problem. This result may be of independent interest.

To construct a reduction in the opposite direction, i.e. a reduction of the $P_{\mathbb{B}}$ -realizability problem to the chamber hitting problem, we settle some technical difficulties. It turns out that a natural construction described in Subsection 5.1 gives a reduction of the $P_{\mathbb{B}}$ -realizability problem to the problem of hitting a translate of an integral polyhedral cone represented by generators. To reduce this problem to the problem of hitting a rational cone we use some additional technical tricks and their description is contained in Subsection 5.3. As an intermediate step we consider the case of a simplicial cone (the generating vectors are linear independent). The general case is reduced to the simplicial one using representation of an arbitrary integral cone as a union of a finite set and a finite family of translates of simplicial integral cones (Theorem 6). This result may be regarded as a sort of an integral Caratheodory's theorem. Recall that integral variants of the Caratheodory's theorem are actively studied (see, for instance, [4, 5]) and have important applications to combinatorial optimization.

Finally, in Section 6 we present some decidable and undecidable problems closely related to $P_{\mathbb{B}}$ -realizability problem thus demonstrating its 'borderline' status with respect to computability.

1 Algebraic and combinatorial properties of LRS

Below we remind some algebraic and combinatorial properties of LRS which will be used in the sequel. All needed proofs and references may be found in [14, 2, 7].

1. Let A and h be, respectively, a linear operator and a linear function on a vector space V and let $x, y \in V$. Then the generating function

$$f_{A,x,y}(t) = \sum_{r \geq 0} h(A^r x - y) t^r \quad (2)$$

is rational.

2. The generating function of any LRS is rational and Taylor's series expansion coefficients of an arbitrary rational function at any point in its domain form a LRS.

3. The set of LRS is closed under componentwise addition and multiplication (i.e. under Hadamard product of sequences).

In the standard setting, the input to the Skolem problem is an integer LRS. But if we are interested in signs of the elements of LRS involved only, then there is no difference between integral and rational cases.

Indeed, let a LRS be specified by rational data and let N be the LCM of the denominators of all a_i and b_i . Form an integral LRS

$$y_n = \sum_{i=1}^d N^i a_i y_{n-i} \text{ if } n > d, \quad y_n = N^{n+1} b_n \text{ if } 1 \leq n \leq d. \quad (3)$$

Proposition 1. $N^{n+1}x_n = y_n$ for all n .

Proof. For $1 \leq n \leq d$ the statement follows from the definition (3). For $n > d$ we proceed by induction. Assuming that the statement holds for all $n < k$, we get

$$y_k = \sum_{i=1}^d N^i a_i y_{k-i} = \sum_{i=1}^d N^i a_i (x_{k-i} N^{k-i+1}) = \sum_{i=1}^d N^{k+1} a_i x_{k-i} = N^{k+1} x_k.$$

Thus the statement holds for $n = k$. □

Remark 1. Trivially, using the Euclidean algorithm, the identity

$$\text{LCM}(x, y) = \frac{xy}{\text{LCF}(x, y)}$$

and by Proposition 1 we can compute y_n from x_n in time polynomial in the length of the input data. So for algorithmic problems, which are concerned with the signs of LRS elements only, integral and rational cases are equivalent w.r.t. polynomial reductions.

Hereinafter we assume that in algorithmic problems an LRS is represented by the list of coefficients a_i, b_j written in binary.

Some LRS are solutions of enumeration problems and for this reason are nonnegative. For instance, the family of the so called \mathbb{N} -rational sequences coincide with the set of generating functions of regular languages (see, [2]). In the sequel \mathbb{N} -rational sequences are called *regular sequences*.

To be more precise, any deterministic automaton A over the alphabet Σ defines LRS s_n , where

$$s_n(A) = \#\{w : \text{word } w \text{ is accepted by } A \text{ and } |w| = n\}. \quad (4)$$

Exactly the sequences of the type (4) will be called regular.

LRS are not regular as they may have negative elements, but nevertheless the following statement holds.

Theorem 1 ([11, Cor. 8.2]). *Any LRS is a difference of two regular sequences.*

2 Orbits of linear maps and LRS

In this section we state formally the orbit description problem and the related chamber hitting problem, and indicate relations of these problems to LRS.

Let h_1, \dots, h_m be a family of affine functions defined on a coordinate space \mathbb{Q}^d . The sign patterns of these functions induce a partition of \mathbb{Q}^d into chambers. Formally, let $s \in \{\pm 1, 0\}^m$. Then a *chamber* is a set

$$\{x \in \mathbb{Q}^d : \text{sign}(h_i(x)) = s_i \text{ for } 1 \leq i \leq m\},$$

where $\text{sign}(t)$ is a standard sign function

$$\text{sign}(t) = \begin{cases} 1, & \text{for } t > 0, \\ 0, & \text{for } t = 0, \\ -1, & \text{for } t < 0. \end{cases}$$

Orbit description problem (ODP).

INPUT: a square matrix Φ of order d ; d -dimensional vector x_0 ; a family of affine functions h_1, \dots, h_m on \mathbb{Q}^d and a set of sign patterns $s_1, \dots, s_r \in \{\pm 1, 0\}^m$.

OUTPUT: ‘yes’ if any point of the orbit $\text{Orb}_\Phi x_0$ falls into the union of chambers $H_{s_1} \cup \dots \cup H_{s_r}$, and ‘no’ otherwise.

Remark 2. We assume that matrices, vectors and affine functions in the ODP and all related problems are represented by the component lists, where the components are written in binary.

Chamber hitting problem (CHP) has the same input as the ODP, but there is one sign pattern s only.

The output of CHP is ‘yes’ if the orbit $\text{Orb}_\Phi x_0$ intersects the chamber H_s and ‘no’ otherwise.

Proposition 2. *CHP is Turing-equivalent to ODP.*

Proof. To reduce the CHP to the ODP we take the complement in the set $\{\pm 1, 0\}^m$ to the sign pattern of the chamber in the input of an instance of the CHP and fix all other input parameters. We obtain an instance of the ODP that outputs ‘yes’ iff the instance of the CHP outputs ‘no’.

The reduction in the opposite direction is proved analogously. For any chamber not included in the input list of an instance of the ODP we solve the corresponding CHP. All the answers are ‘no’ iff the instance of the ODP reports ‘yes’. \square

Remark 3. It is rather obvious that the CHP is recursively enumerable (belongs to the class Σ^1 of the arithmetic hierarchy) and the ODP is co-enumerable (belongs to the class Π^1). The proof of Proposition 2 shows that these problems are complementary in a broad sense.

Another pair of complementary problems is the chamber description problem (the ODP restricted to the case of one chamber) and the problem of hitting the complement to a chamber.

We do not know reducibilities between the chamber description problem and the CHP.

Proposition 3. *The nonnegativity problem is equivalent to the ODP restricted to one linear function and a nonstrict inequality (the chambers H_0, H_{+1}).*

The Skolem problem is equivalent to the CHP restricted to one linear function and equality (the chamber H_0).

Proof. As is well known the LRS (1) is related to a linear operator A in the d -dimensional coordinate space given in matrix notation by

$$A = \begin{pmatrix} a_1 & a_2 & \dots & a_{d-1} & a_d \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & 0 \end{pmatrix} \quad (5)$$

It is easy to check by induction that

$$A^k(b_d, \dots, b_1)^T = (x_{k+d}, x_{k+d-1}, \dots, x_{k+1})^T.$$

Thus, the Skolem problem for the LRS (1) is reduced to the CHP with $\Phi = A$, $x_0 = (b_d, \dots, b_1)^T$, $h_1 = x_d$ and the chamber H_0 .

Analogously, the nonnegativity problem is reduced to the ODP with the same input as above and two chambers H_0, H_{+1} .

The reductions in the opposite direction use standard facts about LRS listed in section 1. Indeed, the generating function $\sum_n h(A^n x_0) t^n$ is rational. Hence, the sequence $h(A^n x_0)$ is an LRS and both reductions follow. \square

We don't know whether the general ODP and CHP can be reduced to, respectively, the nonnegativity problem and the Skolem problem, but we can indicate particular cases when such reductions do exist.

Union of subspaces hitting problem (SHP).

INPUT: a square matrix Φ of order d ; a d -dimensional vector x_0 ; a family of linear functions h_{jk} on \mathbb{Q}^d , $1 \leq j \leq m$, $1 \leq k \leq r_j$.

OUTPUT: 'yes' if the orbit $\text{Orb}_\Phi x_0$ intersects the union of affine subspaces

$$\bigcup_{j=1}^m \{x : h_{j1}(x) = h_{j2}(x) = \dots = h_{jr_j}(x) = 0\}$$

and 'no' otherwise.

Lemma 1. *The SHP is reducible to the Skolem problem.*

Proof. It follows from properties of LRS listed in Section 1 that the sequences

$$\varphi_n(j, k) = h_{jk}(\Phi^n x_0), \quad j = 0, \dots, m,$$

are LRS. But as LRS are closed under componentwise sum and product, the sequence

$$\varphi_n = \prod_{j=0}^m \sum_{k=1}^{s_j} \varphi_n(j, k)^2$$

is also LRS.

Note that taking the input of the SHP we can algorithmically compute the representation of φ_n in the form (1). And the reduction of the SHP to the Skolem problem follows as $\varphi_n = 0$ iff for some j it holds

$$\varphi_n(j, 1) = \varphi_n(j, 2) = \cdots = \varphi_n(j, s_j) = 0.$$

□

Remark 4. Note that m -reducibility in the proof above may not be a polynomial one as the degree of the sequence φ may be exponential w.r.t. the initial degree d . But the answer to a general SHP is disjunction of answers to separate subspace hitting problems. And for this particular case the reducibility described in the proof is polynomial. Hence, we conclude that the SHP can be solved in polynomial time with the Skolem problem oracle. In other words, the SHP is polynomial time Turing reducible to the Skolem problem.

The famous Skolem-Mahler-Lech theorem [2, 7, 15] asserts that the set of zeroes of a LRS consists of a finite set and a union of a finite number of arithmetical progressions.

The proof of Lemma 1 implies that the same is true for the SHP.

Corollary 1 (Skolem-Mahler-Lech theorem for SHP). *The set of those k for which $\Phi^k x_0$ falls into the union of subspaces V_1, \dots, V_m , is a union of a finite set and a finite number of arithmetical progressions.*

Polyhedron localization problem (PLP) is a particular case of ODP when the family of chambers forms a closed convex polyhedron (i.e. a solution set of a system of nonstrict linear inequalities).

Lemma 2. *The PLP is reducible to the nonnegativity problem.*

Proof. W.l.o.g. we assume that the sign pattern of the interior of a polyhedron is $(+1, \dots, +1)$ and sign patterns for the faces of the polyhedron are obtained by replacing some 1's by 0's. Indeed, all negative components in a sign pattern can be removed by a sign change. And all zero components in the sign pattern of the interior of a polyhedron can be removed after checking that the orbit falls into (affine) subspace resulting from the solution of a system of linear equations (corresponding to zero components of the sign pattern). The last problem is certainly decidable as the affine hull of the orbit coincides with the affine hull of the first $d + 1$ orbit points.

Now rearrange the sequences $\varphi_n(j) = h_j(\Phi^n x_0)$ into one sequence φ_n by consequently putting the elements of $\varphi_n(j)$ into places with an index having residue j modulo m , i.e. $\varphi_{sm+j} \stackrel{def}{=} \varphi_s(j)$, $s \geq 0$, $0 < j < m$. The resulting sequence φ_n is an LRS. Indeed, its generating function $f(t)$ is rational as it satisfies the identity

$$f(t) = \sum_{j=1}^m t^j f_j(t^m),$$

where for any $j = 1, \dots, m$, a function $f_j(t)$ is a generating function for $\varphi_n(j)$. Hence, $f(t)$ is a finite sum of rational functions. □

The Skolem problem is the weakest of all problems mentioned above. To decide the Skolem problem it is enough to enumerate the family of LRS with nonnegative elements or the family of the PLP instances with ‘yes’ answers (due to m -reducibility in Lemma 2).

The following proposition belongs to folklore.

Proposition 4. *If the family of LRS with nonnegative coefficients is recursively enumerable then the Skolem problem is decidable.*

Proof. If a sequence x_n is a LRS then the sequence $x_n^2 - 1$ is also a LRS as the family of all LRS is closed under componentwise sum/difference and product, see, Section 1. But the sequence $x_n^2 - 1$ has nonnegative elements iff $x_n \neq 0$ for all n . To complete the proof we apply E. Post theorem [13]. We enumerate all LRS with nonnegative coefficients and compare them to $x_n^2 - 1$. In parallel we enumerate all LRS having zeroes and compare them to $x_n^2 - 1$ also. One of these enumerations should eventually stop. \square

3 CHP and regular languages

In this section we relate the Skolem problem to some properties of regular languages. To be more exact we define problems of *regular realizability*: whether a given regular language contains a word of a particular kind. It is convenient to describe the problem of realizability as follows. Let L be a language in a finite alphabet Σ . Informally, L encodes a *property* that is checked. The input to the problem of L -realizability is a description of some regular language $R \subseteq \Sigma^*$ and we are asked whether the intersection $L \cap R$ is nonempty.

Remark 5. Depending on a way of presentation of a regular language we obtain different in general variants of L -realizability problem. We assume that R is given via deterministic automaton accepting it. These clarification is of course immaterial if we are interested in decidability only, but may be important for complexity estimates.

The *permutation filter* $P_{\mathbb{B}}$ is a language over the alphabet $\{\#, 0, 1\}$ of all words $\#w_1\#w_2\#\dots w_N\#$, where $N = 2^n$, $n \geq 1$, and the set $\{w_i\}$, $i = 1, 2, \dots, N$ consists of all binary words of the length n .

Words from $P_{\mathbb{B}}$ are called *permutation words*. Informally, a permutation word corresponds to a permutation of a set of all binary words of some fixed length n . This length is called the *block rank* of a permutation word.

Theorem 2. *CHP and $P_{\mathbb{B}}$ -realizability problem are Turing equivalent.*

To prove the theorem we construct reductions in both directions, but as a matter of fact, their complexities are essentially different.

In the next section we polynomially reduce the CHP to the $P_{\mathbb{B}}$ -realizability problem and obtain as a corollary NP -hardness of the last problem. This follows from the fact that the Skolem problem is NP -hard [3] and according to proposition 3 it is polynomially reducible to CHP.

Remark 6. A different proof of NP -hardness of the $P_{\mathbb{B}}$ -realizability problem can be derived from the properties of periodic filters, see [18].

The reduction in the opposite direction is shown in Section 5 but it takes superexponential time.

4 Polynomial reduction of the CHP to the $P_{\mathbb{B}}$ -realizability problem

We briefly sketch the idea of the reduction. First, using (2) we associate a LRS with any linear constraint (equality or inequality) in the description of a chamber. Then by proposition 1 we convert this LRS into an integral one. According to Theorem 1 this LRS can be represented as a difference of two regular sequences. It turns out that this construction is not enough to establish polynomial reduction and we elaborate it slightly and represent the integral LRS involved as a difference of two regular sequences on some explicit arithmetic progression of its indices. The last representation can be computed in polynomial time. Next we find a special automaton that compares numbers of words of specified length in two regular languages provided the input is a permutation word. This trick gives a reduction of the Skolem problem to the $P_{\mathbb{B}}$ -realizability problem. To finish the proof we need an additional construction converting several automata described above into one.

We proceed to the proof.

At first note that for any square matrix A of order d , d -dimensional vector x_0 and a linear function h having rational coefficients there is a polynomial time algorithm to recover the coefficients of LRS for the sequence $h(A^n x_0)$. Indeed, the degree of the LRS involved does not exceed the order of the matrix and, hence, the coefficients of LRS and the initial data can be determined if we solve the corresponding system of linear equations for the first $2d$ elements of the sequence.

In this section we assume that all LRS involved have integer coefficients and integral initial data. This is possible due to Proposition 1 as we are interested in signs of the expressions $h(A^n x_0)$ only.

Next we express a LRS as a sum of weights of the walks in a digraph with fixed start and finish vertices. Let $\Gamma(V, E)$ be a digraph (loops and parallel edges allowed) and let $c: E \rightarrow \mathbb{Z}$ be a weight function for edges of Γ . A weight of a walk is a product of weights over the edges of the walk.

Lemma 3. *There exists a polynomial time algorithm that takes an integer LRS $\{x_n\}$ and outputs a weighted digraph G_x with two marked vertices s and f , such that the sum of weights over all walks of the length n starting at the vertex s and finishing at the vertex f equals x_n for all $n \geq 1$.*

Proof. Take a LRS (1) and construct a digraph G_x as follows. (See Fig. 1.)

Connect the start vertex s and the finish vertex f by d vertex-disjoint (except for the terminals) paths P_1, \dots, P_d such that the edge length of P_i is i . Add d vertex-disjoint (except for the terminal f) directed cycles C_1, \dots, C_d , passing through the vertex f .

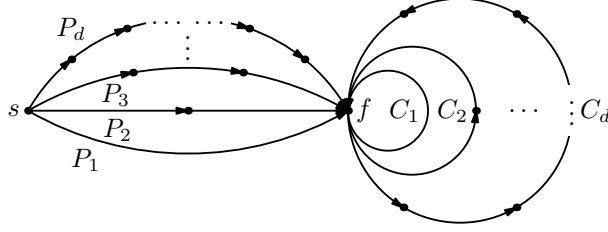


Figure 1: A graph for a LRS

The edge length of C_i is also i . The cycles C_i have no common vertices with the paths P_j except for the vertex f .

Now set edge weighting of G_x . All edges of G_x except for the first edges of all paths P_i and cycles C_j has weight 1. Set the weight of the first (outgoing from s) edge on the path P_i to p_i , $i = 1, \dots, d$, and set the weight of the first (outgoing from f) edge on the cycle C_i to q_i , $i = 1, \dots, d$.

Let z_n be the sum of weights over all walks of the length n from s to f . There is a recurrence on z_n . Note that for $n \leq d$ there exists a walk of the length n from s to f along a path of the digraph and for $n > d$ any walk of the length n is obtained from a shorter one by adding a cycle. Thus

$$z_i = p_i + \sum_{j=1}^{i-1} q_{i-j} z_j, \quad \text{for } 1 \leq i \leq d;$$

$$z_n = \sum_{j=1}^d q_j z_{n-j}, \quad \text{for } n > d.$$

To ensure equalities $z_n = x_n$ set

$$q_j = a_j.$$

For $1 \leq i \leq d$ equalities $z_i = b_i$ imply

$$\begin{aligned} p_1 &= b_1; \\ p_2 &= b_2 - q_1 b_1; \\ p_3 &= b_3 - q_1 b_2 - q_2 b_1; \\ &\dots \end{aligned}$$

To complete the proof note that all computations above take a polynomial time. \square

Lemma 4. *There exists a polynomial time algorithm that takes positive integers n (in binary) and k (in unary) and outputs a digraph, a start vertex s and a finish vertex f such that the number of paths of the length k from s to f equals n provided $k > \log_2 n$.*

Proof. Let L_t be a doubled directed path having $t+1$ vertices $\{1, 2, \dots, t+1\}$ and pairs of parallel edges $(i, i+1)$ between any consecutive vertices i and $i+1$, $i = 1, \dots, t$. Clearly, there are exactly 2^t walks of the length t between vertices 1 and $t+1$.

A *thread* L_j^l is a digraph obtained from L_j by attaching a directed path of the length $k - j$ to its finish vertex $j + 1$. The start and the finish vertices of a thread are, respectively, the start vertex of L_j and the finish vertex of a path.

Let $\{j_1, \dots, j_l\}$ be the positions of nonzero bits in the binary representation n . Take the threads L_{j_i} , $i = 1, \dots, l$ and identify their start vertices and, respectively, their finish vertices into global start vertex s and global finish vertex f . By construction there are exactly n paths of the length k from s to f in the resulting graph.

The case of $n = 11$, $k = 4$ is illustrated in Fig. 2.

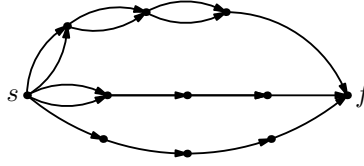


Figure 2: Construction of Lemma 4 for $n = 11$, $k = 4$

Evidently the algorithm is polynomial. □

Now we are able to prove a variation of Theorem 1.

Theorem 3. *There exists a polynomial time algorithm that takes an integer LRS x_n and outputs an integer ℓ that is polynomially bounded in the input length (i.e. the total binary length of all coefficients and the initial data of the LRS) and two deterministic automata A, B over the alphabet $\{0, 1\}$ such that $x_n = s_{\ell n}(A) - s_{\ell n}(B)$ for all $n \geq 1$.*

Proof. The first step is described in the proof of Lemma 3. As a result we obtain a digraph H .

Then we choose integers $M > \log_2 \max\{|a_1|, \dots, |a_d|, |b_1|, \dots, |b_d|\}$ and $k > \log_2(2Md)$. Now the integer ℓ to output is set to $M + k$.

At the next step we apply Lemma 4 and transform the digraph H to a digraph satisfying two additional properties:

- 1) the fan-out of each vertex is 2;
- 2) the absolute value of each weight is 1.

The transformation consists of the following steps.

1. For each integer a_i (b_i) apply Lemma 4 to construct a digraph A_i (B_i) having exactly $|a_i|$ ($|b_i|$) paths of the length M from the start to the finish.

2. Replace the first edges of all cycles C_i (respectively, of all paths P_i) of the graph H by pasting in the graphs A_i (respectively, B_i). Namely, the start vertex of a graph A_i (B_i) is identified with the initial vertex of an edge and the finish vertex is identified with the end vertex of the edge.

Other edges of the graph H are replaced by directed paths of the length ℓ .

Fan-outs of all vertices in the resulting digraph H' are at most 2 except for the start vertex s and for the finish vertex f . Note that fan-outs of the start and the finish vertices are at most $2Md$.

3. Attach to the start and to the finish vertex of the graph H' rooted directed binary trees of depth k having $2Md$ leaves. Replace the starting vertex of each edge outgoing

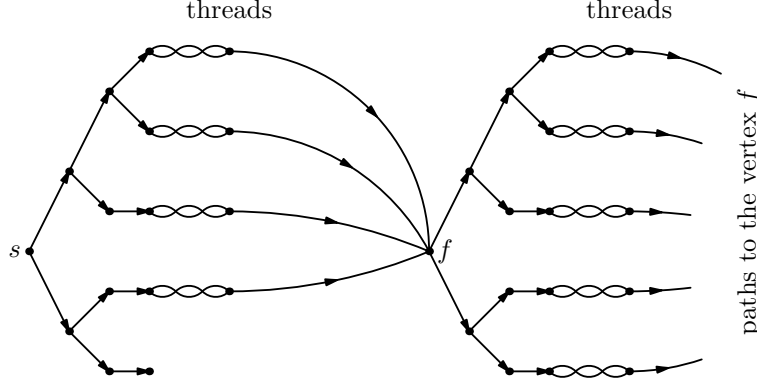


Figure 3: A graph H''

the start vertex (the finish vertex) by a leaf of the corresponding tree in such a way that the fan-out of each vertex in the resulting graph H'' is at most 2.

As a result we obtain a modified digraph H'' (see Fig. 3). It has vertices with the fan-out less than 2. Add an auxiliary vertex u and edges to the u from all vertices with the fan-out less than 2 in the digraph H' to make all fan-outs equal 2. Then add two loops at the vertex u . After these operations we obtain a digraph G .

Now we assign weights to the edges of the digraph G . The weight of an edge ingoing to the start vertex of a thread of a digraph A_i (B_i) coincides with the sign of the corresponding integer a_i (b_i). All other edges have weights 1.

It is clear from the construction that in the digraph G the sum of weights over all walks of the length ℓn from s to f equals x_n .

Now we construct automata A and B over the alphabet $\{0, 1\}$. For both of them the state set is $\{\pm 1\} \times V(G)$. Choose a labelling of the edges of the digraph G by 0 and 1 such that for each vertex edge labels of the two outgoing edges are different. (Recall that the fan-out of any vertex of G is 2.)

Let's describe the automaton A .

The initial state of the A is $(+1, s)$ and the only accepting state is $(+1, f)$. Reading a symbol α in the state (σ, w) the automaton A goes to the state (σ', w') if the edge (w, w') in the digraph G has label α . If the edge (w, w') has positive weight then $\sigma' = \sigma$. Otherwise, $\sigma' = -\sigma$. The rule is illustrated in Fig. 4.

The automaton B differs from the automaton A in the accepting state only. The accepting state of B is $(-1, f)$.

It is clear that the lengths of all words accepted by the automata A and B are multiples of ℓ .

Note that both automata can be constructed from the initial LRS in polynomial time.

To finish the proof we note that x_n equals the difference between the number of words of the length ℓn accepted by the automata A and B . Indeed, the walks of negative weight correspond to the words accepted by the B while the walks of positive weight correspond to the words accepted by the A . \square

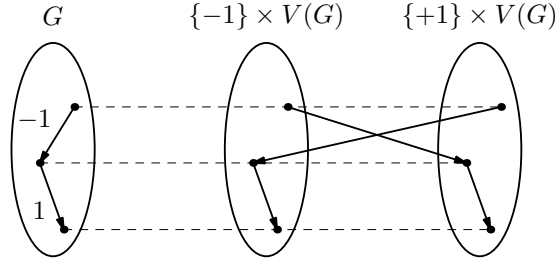


Figure 4: Counting the sign

Now we are ready to present a polynomial reduction of the Skolem problem to the $P_{\mathbb{B}}$ -realizability problem.

Theorem 3 implies that for the reduction it is sufficient to build up an algorithm running in polynomial time such that takes a pair of automata A, B over the alphabet $\{0, 1\}$ and an integer ℓ in unary and outputs the description of an automaton C such that $L(C) \cap P_{\mathbb{B}} \neq \emptyset$ iff $s_{\ell n}(A) = s_{\ell n}(B)$ holds for some n .

Let informally explain how to construct such an automaton. The automaton C is made of two automata C' and C'' by the product construction. The automaton C accepts iff C' and C'' accept.

Recall that we are interested in operation of automata on permutation words, i.e., words of the type $\#w_1\#w_2\#\dots\#w_k\#$, where $w_i \in \{0, 1\}^*$, $i = 1, \dots, k$.

The block rank (the length of each w_i) should be a multiple of ℓ . This last condition is verified by the automaton C'' . It counts the length of w_i modulo ℓ and rejects in the case of a nonzero residue. Otherwise it accepts.

The automaton C' starts from an accepting state q_0 and reads the current block word w_i separated by the delimiters $\#$.

If w_i belongs to $L_{AB} = L(A) \setminus L(B)$ then C' reads the next block word w_{i+1} and checks whether it belongs to $L_{BA} = L(B) \setminus L(A)$. If the last check fails then C' rejects, otherwise it returns to the state q_0 .

If w_i belongs to L_{BA} then C' rejects.

If w_i belongs to $L_{\sim} = (L(A) \cap L(B)) \cup \overline{L(A) \cup L(B)}$ then C' passes to a new accepting state q_1 .

Starting from the state q_1 the automaton C' reads the current block word w_i . If the block belongs $L_{AB} \cup L_{BA}$ then C' rejects. Otherwise it returns to the state q_1 .

The structure of the automaton C' is pictured in Fig. 5.

Let prove that C gives the required reduction of the Skolem problem to the $P_{\mathbb{B}}$ -realizability problem.

First we must show that if an instance of the Skolem problem has a positive answer then C accepts at least one word from the permutation filter.

Indeed, assume that some x_n vanishes. Then $s_{\ell n}(A) = s_{\ell n}(B)$ and there is a one-to-one correspondence between the words of the length ℓn in L_{AB} and the words of the length ℓn in L_{BA} . A required permutation word W is obtained by arranging corresponding words in pairs (other words of the length ℓn are arranged arbitrary after all pairs). By construction C accepts W .

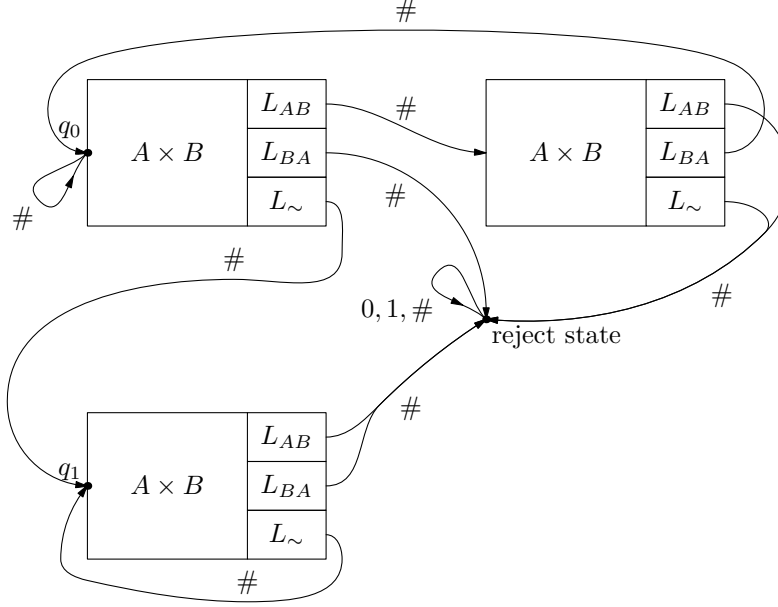


Figure 5: The structure of the automaton C' . The initial state is q_0 . The only accepting states are q_0, q_1

On the other hand, assume that C accepts some word W from the permutation filter. It follows from the construction that the block rank of W is a multiple of ℓ . We can write W in the form W_1W_2 . Here W_1 is prefix of W that is a concatenation of $\#w_i\#w_{i+1}$, where $w_i \in L_{AB}$ and $w_{i+1} \in L_{BA}$. And W_2 is a suffix of W that is concatenation of $\#w_i$, where $w_i \in L_{\sim}$. Thus, $|L(A) \setminus L(B)| = |L(B) \setminus L(A)|$. It means that $x_n = 0$ and the instance of the Skolem problem has a positive answer.

Clearly the size of the automata C is polynomial in the size of the LRS. This finishes the proof of the reduction.

From the results of this section and the results from [3] we get an immediate corollary.

Theorem 4. *The problem of $P_{\mathbb{B}}$ -realizability is NP-hard.*

Now we extend the previous result to the general chamber hitting problem.

The construction of the automaton C can be easily modified to accept a permutation word satisfying the condition $s_{\ell n}(A) < s_{\ell n}(B)$. In the notation above to be accepted the suffix W_2 of a permutation word should contain at least one more occurrence of a word from $L(B) \setminus L(A)$.

The counter part of the modified automaton $C_{<}$ does not change. The structure of the automaton $C'_{<}$ is pictured in Fig. 6.

It is evident that the size of the modified automaton $C_{<}$ is upperbounded by the size of the automaton C up to a constant factor.

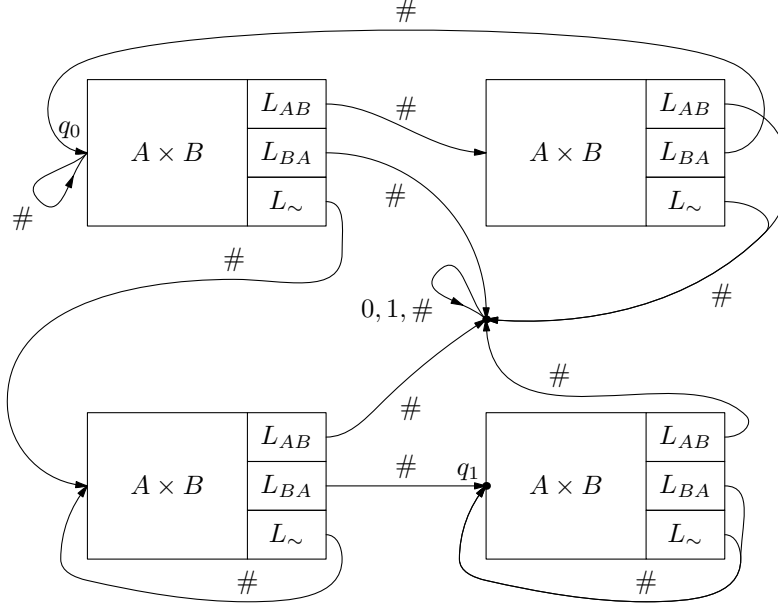


Figure 6: The structure of the automaton $C'_<$. The initial state is q_0 . The only accepting state is q_1

To complete a reduction of the chamber hitting problem to the $P_{\mathbb{B}}$ -realizability problem we construct an automaton that checks several conditions of the form $s_{\ell n}(A) = s_{\ell n}(B)$ ($s_{\ell n}(A) < s_{\ell n}(B)$).

Repeating the previous argument let's construct for each constraint $h_i(\Phi^n x_0) < 0$ (or $h_i(\Phi^n x_0) = 0$) from the description of the chamber a pair of automata A_i , B_i and an integer ℓ common to all pairs A_i , B_i such that a constraint $h_i(\Phi^n x_0) < 0$ ($h_i(\Phi^n x_0) = 0$) holds iff an inequality $s_{\ell n}(A) < s_{\ell n}(B)$ (an equality $s_{\ell n}(A) = s_{\ell n}(B)$) holds. The main difficulty here stems from the condition that the block ranks ℓn of all permutation words certifying the corresponding constraints in the reducibility **must be equal**. To solve this problem we use the following arguments.

At first, choose the same integer ℓ for all pairs of automata. The required value of ℓ is 1 plus the three times the maximum of all logarithms of all data for all LRS involved. Then increase to $\ell/2$ the values of the parameters k and M from the proof Theorem 3. It can be done by attaching paths to threads of the graphs constructed in Lemma 4.

Next step is to construct for each pair A_i , B_i , where $1 \leq i \leq m$ and m is the number of constraints of CHP, the automaton C'_i that certifies the inequality $s_{\ell n}(A) < s_{\ell n}(B)$ (or the equality $s_{\ell n}(A) = s_{\ell n}(B)$) as described above.

If m is not a power of 2 then by definition an automaton C'_i , where $m < i \leq 2^p$, $p = \lceil \log_2 m \rceil$, is a dummy automaton accepting all words.

Now all automata C'_i are combined into an automaton C that checks all conditions $s_{\ell n}(A) ? s_{\ell n}(B)$ for some length ℓn .

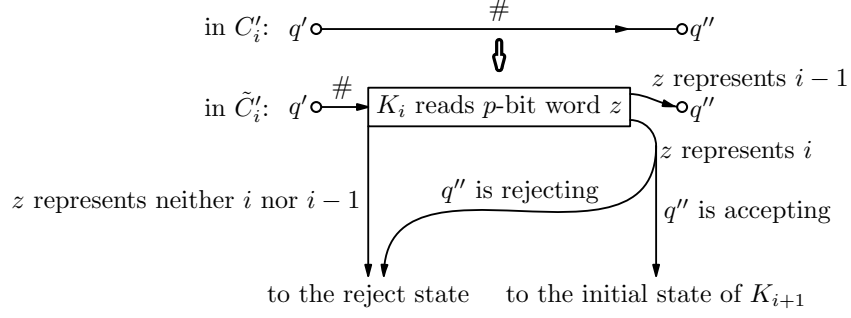


Figure 7: Modification of a $\#$ -transition in an automaton C'_i

The combined automaton has a product form $\tilde{C} = \tilde{C}' \times \tilde{C}''$ and accepts words $\#w_1\#w_2\#\dots w_N\#$ from the permutation filter that satisfy the following requirements.

- The block rank is $p + \ell n$ for some n (in the sequel we denote $w_i = u_i v_i$, where $|u| = p$).
- Prefixes u_i form a nondecreasing sequence w.r.t. the lexicographic order.
- For each $1 \leq r \leq m$ let z be the binary representation of $r - 1$ of the length p . Form a subword $\#w_i\#w_{i+1}\#\dots\#w_{i+j}\#$ that consists of all block words with the prefix z . Then the word $\#v_i\#v_{i+1}\#\dots\#v_{i+j}\#$ is accepted by the automaton C'_r .

The first requirement is verified by a counter \tilde{C}'' . The second and the third requirements are verified by an automaton \tilde{C}' , which is a sort of a concatenation of modified automata \tilde{C}'_i . In a modified automaton \tilde{C}'_i each $\#$ -transition from a state q' to a state q'' is changed by a $\#$ -transition from q' to the initial state of a copy of an auxiliary automaton K_i . This automaton reads a word z of the length p and compares it with p -length binary representations of $i - 1$ and i . If z represents $i - 1$ then K_i passes to the state q'' of the automaton \tilde{C}'_i . If z represents i and q'' is an accepting state in the automaton \tilde{C}'_i then K_i passes to the initial state of the automaton \tilde{C}'_{i+1} . Otherwise it rejects. This modification is illustrated in Fig. 7.

The initial state of the automaton \tilde{C}' is the initial state of \tilde{C}'_1 . The accepting states are the initial states of those copies of the auxiliary automaton K_{2^p} that are inserted instead of $\#$ -transitions to accepting states of the automaton \tilde{C}'_{2^p} .

The size of the automaton \tilde{C} is $O(Dm \log m)$, where D is the maximum of the sizes over all automata C_i . Indeed, an auxiliary automaton K_i can be implemented using $O(p) = O(\log m)$ states. So the size of a modified \tilde{C}'_i is increased by a factor $O(\log m)$ and we combine sequentially $2^p = O(m)$ modified automata \tilde{C}'_i . Let prove that the resulting automaton \tilde{C} gives the required reduction of the CHP to the $P_{\mathbb{B}}$ -realizability problem.

Suppose that for some n all constraints in an instance of the CHP are satisfied. Then there exists a permutation of block words of the length ℓn , which satisfies the

accepting requirements of the automaton C_i for each $1 \leq i \leq m$. Extending blocks by prefixes z_i that represent $i - 1$ in binary and arranging permutations in the order of i we obtain the permutation word satisfying all three requirements stated above. Hence this word is accepted by \tilde{C} .

On the other hand, assume that \tilde{C} accepts some word W from the permutation filter. The counter \tilde{C}'' accepts. So the block rank of W is $p + \ell n$. The automaton \tilde{C}' accepts also. It follows from the construction that it must pass through all automata \tilde{C}'_i . There are 2^p possible prefixes of the length p . So the construction of modified automata implies that the sequence of block prefixes of the length p is uniquely determined. Namely, it has a form

$$\underbrace{\overbrace{0000 \dots 0}^{p \text{ bits}}, \dots, \overbrace{0000 \dots 0}^{p \text{ bits}}}_{2^{\ell n} \text{ prefixes}}, \dots, \underbrace{\overbrace{111 \dots 1}^{p \text{ bits}}, \dots, \overbrace{111 \dots 1}^{p \text{ bits}}}_{2^{\ell n} \text{ prefixes}}.$$

Thus acceptance by \tilde{C}' means that suffixes corresponding to each prefix z , where z is a binary representation of $i - 1$, form a permutation of the binary words of the length ℓn and the corresponding permutation word is accepted by the automaton C_i . The latter implies that i th constraint in the instance of the CHP is satisfied. It implies that the answer for the instance of the CHP is positive.

5 Reduction of the $P_{\mathbb{B}}$ -realizability problem to the CHP

The reduction is composed from several intermediate reductions.

At first, we show that the $P_{\mathbb{B}}$ -realizability problem is reducible to the Walk Weight Hitting Problem. The walks in this problem are walks in a digraph.

Let $\Gamma(V, E)$ be a digraph. We assume that the edges of the digraph are colored in s colors from the set $\{1, 2, \dots, s\}$.

For a walk τ we define a *weight* $w(\tau) \in \mathbb{Z}^s$ as an s -dimensional integer vector

$$w(\tau) = (c_1(\tau), c_2(\tau), \dots, c_i(\tau), \dots, c_s(\tau)),$$

where $c_i(\tau)$ equals the number of edges colored in the color i on the walk τ .

Walk Weight Hitting Problem (WWHP)

INPUT: a digraph Γ , an s -coloring of the edges of the Γ , two vertices a, b of Γ , an integer matrix Φ of order $s \times s$ and an integer vector x_0 of dimension s .

OUTPUT: ‘yes’ if the orbit $\text{Orb}_{\Phi} x_0$ intersects a set of weights of the walks from the vertex a to the vertex b and ‘no’ otherwise.

Lemma 5. *The $P_{\mathbb{B}}$ -realizability problem is Turing reducible to the WWHP problem.*

The proof of Lemma 5 is presented in Subsection 5.1.

Then we will reduce the WWHP to the Integer cone Hitting Problem. An *integer cone* $\mathbb{N}(v_1, \dots, v_r)$ is the set of vectors

$$\sum_{i=1}^r a_i v_i, \quad a_i \in \mathbb{N},$$

where $v_i \in \mathbb{Z}^d$. We denote the set of nonnegative integers by \mathbb{N} .

Integer cone Hitting Problem (IHP)

INPUT: a square matrix Φ of order d ; a d -dimensional vector x_0 ; a family of vectors $v_i \in \mathbb{Z}^d, i = 0, 1, \dots, r$.

OUTPUT: ‘yes’ if the orbit $\text{Orb}_\Phi x_0$ intersects the translate of the integer cone $v_0 + \mathbb{N}(v_1, \dots, v_r)$ and ‘no’ otherwise.

Lemma 6. *The WWHP is Turing reducible to the IHP.*

The proof of Lemma 6 is presented in Subsection 5.2.

Next step is to reduce the IHP to the Polyhedral cone Hitting Problem. This last intermediate problem is stated as follows.

Polyhedral cone Hitting Problem (PHP)

INPUT: a square matrix Φ of order d ; a d -dimensional vector x_0 ; a family of linear functions h_j on \mathbb{Q}^d ; a shift vector $v_0 \in \mathbb{Q}^d$.

OUTPUT: ‘yes’ if the orbit $\text{Orb}_\Phi x_0$ intersects the translate $v_0 + K$ of the closed polyhedral cone

$$K = \{x \in \mathbb{Q}^d : h_j(x) \geq 0\}$$

and ‘no’ otherwise.

The PHP problem is Turing-reducible to the ODP problem. The proof is similar to the proof of Proposition 2.

So, the total chain of reductions is

$$P_{\mathbb{B}}\text{-realizability} \leq_T \text{WWHP} \leq_T \text{IHP} \leq_T \text{PHP} \leq_T \text{ODP} \leq_T \text{CHP}.$$

As mentioned above the last two reductions are based on Proposition 2. So, to complete the reduction we should prove the following theorem.

Theorem 5. *The IHP problem is Turing reducible to the PHP problem.*

The proof of Theorem 5 is contained in Subsection 5.3.

5.1 Proof of Lemma 5

We start from some preliminary work.

Let R be a regular language over the alphabet $\{0, 1, \#\}$ and let A be a deterministic automaton with the state set Q accepting the language R . Our goal is to check whether $R \cap P_{\mathbb{B}} \neq \emptyset$. Let pass to the transition monoid and express this condition in terms of three maps $f_0, f_1, f_{\#}$ of the set Q into itself induced by reading respective symbols.

Define $f(w)$, where $w = w_1 w_2 \dots w_\ell \in \{0, 1\}$, as

$$\prod_{i=1}^{\ell} f_{w_{\ell-i+1}}.$$

We denote the initial state of the automaton by q_s and the accepting set by Q_a . The reduction algorithm checks for each accepting state $q_f \in Q_a$ whether it can be reached from the initial state q_s after reading some word from the permutation filter. Formally this condition means that for some word $\#w_1\#w_2\#\dots w_N\# \in P_{\mathbb{B}}$ we have

$$q_f = \left(\prod_{i=0}^{N-1} f_{\#} f(w_{N-i}) \right) f_{\#} q_s. \quad (6)$$

It is important that to verify the condition (6) we do not need to know the sequence w_i . It is sufficient to compute for each map $g \in Q^Q$ the number $\nu_n(g)$ of its representations in the form $f(w)$, where $w \in \{0, 1\}^n$ (recall that $N = 2^n$). Then the condition (6) is rewritten as

$$q_f = \left(\prod_{i=0}^{N-1} f_{\#} g_i \right) f_{\#} q_s, \quad (7)$$

where each map $g \in Q^Q$ occurs exactly $\nu_n(g)$ times in the sequence g_i .

It turns out that the integers $\nu_n(g)$ can be expressed as the coordinates of the vectors taken from the orbit of a linear map.

In \mathbb{Q} -vector space $\mathbb{Q}(Q^Q)$ equipped with the basis $\{e(f)\}$ indexed by maps $f: Q \rightarrow Q$ we define a linear map by the action on the basis vectors

$$\Phi e(f) = e(f f_0) + e(f f_1). \quad (8)$$

(Recall that map composition is taken from the right to the left.)

Proposition 5. $\nu_n(g)$ equals the $e(g)$ -coordinate of the vector $\Phi^n e(\text{id})$ w.r.t. the basis $\{e(f)\}$. Here id is the identity map.

Proof. By induction on n . The case of $n = 0$ is trivial. If the statement holds for $n = k - 1 \geq 0$ then it holds for $n = k$:

$$\begin{aligned} \Phi^k e(\text{id}) &= \Phi \sum_{g \in Q^Q} \nu_{k-1}(g) e(g) = \sum_{w \in \{0,1\}^{k-1}} \Phi e(f(w)) = \\ &= \sum_{w \in \{0,1\}^{k-1}} (e(f(w) f_0) + e(f(w) f_1)) = \sum_{w \in \{0,1\}^{k-1}} (e(f(w 0)) + e(f(w 1))) = \\ &= \sum_{w \in \{0,1\}^k} e(f(w)) = \sum_{g \in Q^Q} \nu_k(g) e(g). \end{aligned}$$

□

Now we are going to describe the condition (7) in terms of walk weights for a suitable graph.

For this purpose we will use the Cayley graphs for monoids.

Let $G = \{g_1, \dots, g_m\} \subseteq Q^Q$ be a set of maps. It generates a monoid M (a monoid operation is a map composition, recall that by definition a monoid contains also the identity map.). By definition the vertices of the Cayley graph Γ_G of the monoid are elements of the monoid M and (directed) edges have the form $(h, g_i h)$ for $h \in M$, $g_i \in G$. Note that the edges of the Cayley graph are naturally colored by elements of G . The Cayley graph of a monoid may contain parallel edges as the equality $g_i h = g_j h$ for $i \neq j$ is possible for a monoid.

Let M_{01} be a semigroup generated by the maps f_0, f_1 of the automaton A . This semigroup is finite and can be described as the least set of maps from the state set Q to itself that contains the maps f_0, f_1 and is closed w.r.t. multiplication by f_0, f_1 . So, the semigroup M_{01} can be constructed efficiently¹.

In a similar way we define a monoid M generated by maps $f_{\#}f, f \in M_{01}$. This monoid is also constructed efficiently.

Denote by Γ_M the Cayley graph of the monoid M w.r.t. the set of generators $\{f_{\#}f, f \in M_{01}\}$.

It follows from the construction that (7) holds iff there exist an integer n and a walk τ on the digraph Γ_M from the vertex id to a vertex h such that $h(f_{\#}(q_s)) = q_f$ and $(w(\tau))_{f_{\#}g} = \nu_n(g)$ holds for all g . And we conclude that the condition (7) is equivalent to the condition

$$w(\tau) = \Phi^n x_0. \quad (9)$$

Remark 7. Note that the size of the monoid M can be less than $|Q|^{|Q|}$ and Φ acts on $\mathbb{Q}(Q^Q)$. To fix a difference in vector dimensions we extend the coordinates of walk weights by zero values for $e(g)$ such that $g \notin M$.

Thus the reduction algorithm solves with help of a WWHP-oracle several instances of the WWHP indexed by the accepting states q_f and mappings h such that $h(f_{\#}(q_s)) = q_f$.

An instance has the following input data: a digraph is the Cayley graph Γ_M , colors are generators, the initial vertex is the identity map id , the terminal vertex is h , the matrix is defined by (8) and the initial vector is $x_0 = e(\text{id})$.

If the answer is positive for some instance from the described set then the answer in the instance of the $P_{\mathbb{B}}$ -realizability problem involved is also positive due to Proposition 5 and condition (9).

Otherwise, it follows from the definition of the graph Γ_M and Proposition 5 that the answer in the instance of the $P_{\mathbb{B}}$ -realizability problem is negative.

Lemma 5 is proved.

5.2 Proof of Lemma 6

We need the following auxillary lemma.

¹Hereinafter ‘efficiency’ means a mere existence of an algorithm.

Lemma 7. *Let a, b be vertices of a digraph Γ colored in s colors and let $W(a, b)$ be the set of weights of all walks from the vertex a to the vertex b . The set $W(a, b)$ is a finite union of translates of integer cones in \mathbb{Z}^s .*

The list of the cones and the vectors of translation is constructed efficiently.

Proof. Writing a sequence of colors along a walk gives a word in the s -letter alphabet. The collection of such words for all walks from the vertex a to the vertex b forms a regular language.

The weight of a walk is just the Parikh image of the corresponding word. So the statement of the lemma follows from Parikh's theorem [10]. \square

Lemma 7 implies that to check the condition (9) it is sufficient to check several conditions of the form

$$\Phi^n x_0 \in v_0 + W, \quad (10)$$

where v_0 is an integer s -dimensional vector and W is an integer cone in \mathbb{Z}^s . It gives a reduction of the WWHP to the IHP.

5.3 Proof of Theorem 5

Let $W_{\mathbb{Q}} \subseteq \mathbb{Q}^s$ be a rational cone generated by vectors v_1, \dots, v_r . There is an algorithm (see, e.g. [12, §7.2]) for passing to dual polyhedral description of $W_{\mathbb{Q}}$.

The cone $W_{\mathbb{Q}}$ may contain some additional integral points not belonging to integral cone W so that conditions (10) should be modified.

Let us start with a simple case of an integral simplicial cone, when generating vectors v_i are linear independent.

Proposition 6. *If vectors v_1, \dots, v_r are linear independent then representation $x = \sum_i a_i v_i$, $a_i \in \mathbb{N}$ holds if and only if the following two representations hold: $x = \sum_i b_i v_i$, $b_i \in \mathbb{Q}_+$ and $x = \sum_i c_i v_i$, $c_i \in \mathbb{Z}$.*

Proof. The implication \Leftarrow is trivial and the opposite implication \Rightarrow follows from linear independence of vectors v_i , as the equality

$$\sum_i b_i v_i = \sum_i c_i v_i$$

holds iff $b_i = c_i$, $i = 1, \dots, r$. \square

Vector x can be represented in a form $x = \sum_i c_i v_i$, where $c_i \in \mathbb{Z}$, iff x is an element of the subgroup G generated by vectors v_i in \mathbb{Z}^s . And this condition in turn is a conjunction of two conditions. The first condition is simple: x is in a subspace generated by vectors v_i in the vector space \mathbb{Q}^s . To formulate the second condition let extend the set v_i of generators of G to the basis of \mathbb{Z}^s by adding some unit coordinate vectors e_j so that the resulting system formed from v_i and e_j constitute a basis of \mathbb{Z}^s . Such extension is always possible in view of linear independence of v_i . Let \tilde{G} be the group generated by v_i and e_j .

Proposition 7. *Vector x belongs to the subgroup G generated by vectors v_i in the group \mathbb{Z}^s iff x is in the subspace generated by vectors v_i in \mathbb{Q}^s and x belongs to the subgroup \tilde{G} .*

Proof. The implication \Leftarrow is obvious. To prove the opposite implication \Rightarrow assume that $x \in \tilde{G}$. By construction of the subgroup \tilde{G} it follows that

$$x = g + \sum_j x_j e_j,$$

where $g \in G$ and e_j denote those added coordinate unit vectors. Now it follows from linear independence of v_i and e_j that if vector x belongs to the subspace generated by vectors v_i in the coordinate vector space \mathbb{Q}^s then all coordinates x_j are zero. Hence x belongs to the subgroup G . \square

It follows from the propositions 6 and 7 that a vector falls into integral conic hull of vectors v_i if and only if three conditions are fulfilled. The first two conditions are \mathbb{Q} -linear and control whether a vector falls into the cone $W_{\mathbb{Q}}$. The third condition consists in checking that a vector belongs to a fulldimensional lattice in \mathbb{Z}^s (a subgroup of \mathbb{Z}^s having finite index in \mathbb{Z}^s). Checking the first and the second condition for the orbit of a linear map are particular variants of CHP and the third condition holds for a nice family of orbit points.

Proposition 8. *Let $G = \langle v_1, \dots, v_s \rangle \subseteq \mathbb{Z}^s$ be rank s subgroup of \mathbb{Z}^s , let v_0 be an integral vector in \mathbb{Z}^s and let Φ be an integral $s \times s$ matrix.*

Then the set

$$H = \{n : \Phi^n x_0 \in v_0 + G\}$$

is a union of a finite set H_0 and a finite set of arithmetic progressions. Moreover, there is an algorithm to compute H .

Proof. First let G be a diagonal subgroup whose generators are columns of the matrix

$$D = \begin{pmatrix} q_1 & 0 & \dots & 0 \\ 0 & q_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & q_s \end{pmatrix}$$

The shift induced by G is given by

$$x_i \equiv \xi_i \pmod{q_i}, \quad 1 \leq i \leq s. \quad (11)$$

Let q be the least common multiple of q_i . Now calculating $\Phi^k x_0$ modulo q we can find a finite set of exponents k , and the corresponding arithmetic progressions $k \pmod{q}$, satisfying (11). Besides these arithmetic projections H may include only numbers that should be less than initial terms of the progressions found on the previous step, so that we can directly compute this finite set.

The general case is reduced to a diagonal one by reducing the generator matrix M to the Smith normal form (see [12, §4.4]) If G is a subgroup of \mathbb{Z}^s of full rank then Smith normal form is given by

$$M = UDV, \quad (12)$$

where U, V are unimodular matrices and D is a nondegenerate diagonal matrix. V corresponds to column transformations and gives a system of new generators $\tilde{v}_1, \dots, \tilde{v}_s$ of G . U corresponds to row transformations and shows how the vectors e_i of the initial basis can be expressed as a linear combinations of the new basis vectors \tilde{e}_i :

$$(e_1, \dots, e_s) = (\tilde{e}_1, \dots, \tilde{e}_s)U.$$

By (12) the generator matrix \tilde{v}_i of the group G is diagonal in the basis \tilde{e}_i .

To complete a reduction to the diagonal case we should express the matrix Φ and the initial vectors x_0, v_0 in the new basis \tilde{e}_i . \square

Lemma 8. *If vectors v_1, \dots, v_r are linear independent then the IHP is reduced to the PHP.*

Proof. Let us describe an algorithm with PHP-oracle that solves the IHP for instances of linear independent set of vectors v_1, \dots, v_r .

First let find the set H defined in proposition 8. We should check whether condition (10) holds for H . The finite part $H_0 \subseteq H$ can be checked directly. And for each arithmetic progression $n = n_0 + Nk, k = 0, 1, \dots$ in H we compute a vector $x_1 = \Phi^{n_0}$ and a matrix $\Phi_1 = \Phi^N$. Then using PHP-oracle we check whether there exists k such that $\Phi_1^k x_1 - v_0$ hits the rational cone $W_{\mathbb{Q}}$. If all such tests fail we answer ‘no’. Otherwise we answer ‘yes’.

To check correctness of the algorithm involved note that it follows from the propositions 7 and 8 that all tests fail if and only if the orbit $\Phi^n x_0$ either does not intersect the cone $W_{\mathbb{Q}}$ at all or hits it for exponents n such that $\Phi^n x_0 - v_0$ is not in the group G . Hence, the answer in the IHP is negative.

On the other hand, if some test is successful then we find n , such that $\Phi^n x_0 - v_0$ is in the group G and hits the cone $W_{\mathbb{Q}}$. It follows from proposition 6 that the answer in the IHP is positive. \square

Now we discuss the general case. Let

$$W_{\mathbb{N}} = \mathbb{N}(v_1, \dots, v_r)$$

be the set of integral linear combinations of v_i with nonnegative coefficients (the integral cone generated by v_i). Below we describe an algorithm that given any integral cone $W_{\mathbb{N}}$ produces its representation as a union of a finite set W_0 of singular points and a finite set of translates of integral simplicial cones $u_i + \mathbb{N}(v_{i_1}, \dots, v_{i_t})$.

Recall that Caratheodory’s theorem (see, e.g. [12, §7.7]) states that any vector in $\mathbb{Q}_+(v_1, \dots, v_r)$ is a rational nonnegative linear combination of some $\leq s$ (where s is the dimension) vectors from the system v_1, \dots, v_r .

As one can form not more than $\binom{r}{s}$ systems of linear independent vectors out of r vectors, then it is sufficient to consider the case of the intersection of the integral cone $W_{\mathbb{N}}$ and a rational simplicial cone with s generators chosen from the set v_1, \dots, v_r .

Let K be one of such simplicial cones and let $\tilde{v}_1, \dots, \tilde{v}_t$ be the set of its generating vectors. K contains $\mathbb{N}(\tilde{v}_1, \dots, \tilde{v}_t)$, but besides that K may contain some extra integral

points. Now we use the fact that K has *Hilbert basis*: a set of integral vectors u_1, \dots, u_m , such that

$$K \cap \mathbb{Z}^s = \mathbb{N}(u_1, \dots, u_m). \quad (13)$$

In particular, it follows from (13) that

$$K \cap \mathbb{Z}^s = \bigcup_{i=1}^m (u_i + \mathbb{N}(\tilde{v}_1, \dots, \tilde{v}_t)).$$

Recall that one can effectively find Hilbert basis for K as it coincides with the set of integral points of the polytope

$$\{\lambda_1 \tilde{v}_1 + \lambda_2 \tilde{v}_2 + \dots + \lambda_t \tilde{v}_t : 0 \leq \lambda_i \leq 1\}.$$

Some vectors from Hilbert basis belongs to the subgroup \mathbb{Z}^s , generated by v_i , i.e. to the integral hull of v_i . Checking this condition is effective as it is reduced to solving linear Diophantine equations.

Let take one of such vectors $u = \sum_i b_i v_i$ and let B be maximum of modules of the coefficients b_i . Now all points of the intersection of $u + \mathbb{N}(\tilde{v}_1, \dots, \tilde{v}_t)$ and $W_{\mathbb{N}}$ belong to the union of the finite set

$$\{x : x = u + \sum_{i=1}^t a_i \tilde{v}_i, 0 \leq a_i \leq B\}, \quad (14)$$

the integral cone

$$u + (B+1) \sum_i \tilde{v}_i + \mathbb{N}(\tilde{v}_1, \dots, \tilde{v}_t) \quad (15)$$

and $(B+1)t$ sets of the form

$$(u + a\tilde{v}_i + \mathbb{Q}_+(\tilde{v}_1, \tilde{v}_2, \dots, \tilde{v}_{i-1}, \tilde{v}_{i+1}, \dots, \tilde{v}_t)) \cap W_{\mathbb{N}}, \quad 0 \leq a \leq B, 1 \leq i \leq t. \quad (16)$$

Each set in (16) is an intersection of the integral cone $W_{\mathbb{N}}$ and a translate of some rational simplicial cone of the lower dimension $t-1$.

Now we can give a complete description of the algorithm that given an integral cone $W_{\mathbb{N}}$ and a translate of some rational simplicial cone whose generating vectors are taken from the set of generators of $W_{\mathbb{N}}$, finds representation of the intersection of the cones involved as a union of a finite set of singular points W_0 and a finite set of translates of simplicial integral cones.

The procedure is recursive. Applying it to the intersection of $W_{\mathbb{N}}$ and a t -dimensional rational simplicial cone K we at first compute Hilbert basis of K . Then for any vector in Hilbert basis that belongs to the integral hull of vectors v_1, \dots, v_r , we find the sets (14), (15) and (16).

The set (14) is added to the singular part W_0 . The simplicial integral cone (15) is included to the family of integral cones computed on the previous steps and the algorithm is continued recursively by processing all sets (16).

The procedure is finite as the sets (16) are empty for any one-dimensional cone. Hence, the recursion tree has finite height and finite degree at any point.

Thus the following theorem is proved.

Theorem 6. *An integral translate of an integral cone*

$$v_0 + \mathbb{N}(v_1, \dots, v_r)$$

can be represented as a union of a finite set and a finite family of translates of simplicial integral cones. There is an algorithm that finds such a representation from the list of vectors v_0, v_1, \dots, v_r .

Now the main result of this section easily follows.

Proof of theorem 5. Using algorithm from Theorem 6 find representation of the integer cone as a union of a finite set of singular points and finite family of translates of simplicial integral cones. Now for any cone in the family check whether the orbit hits it using PHP-oracle from Lemma 8.

Checking whether or not the orbit hits a point (and hence, any finite set of points) is also reduced to the PHP as a point may be regarded as a translate of the zero cone. \square

6 Decidable and undecidable variants of the regular realizability problem

The Skolem problem is open for almost eighty years. Using slight abuse of language, presently it falls ‘on the border between decidability and undecidability’ [7]. In this section we show that analogous ‘borderline’ pattern holds for a more general $P_{\mathbb{B}}$ -realizability problem and give some decidable and undecidable problems closely related to it. All these problems are problems of regular realizability. The languages specifying the problems consist of binary *block words* separated by the delimiter $\#$. All blocks have the same length (the block rank n). Blocks of a block word form a multiset of binary words of the same length. We will define languages indicating the properties of this *block multiset*.

Now we present decidable examples.

The surjective filter $S_{\mathbb{B}}$ consists of those block words which block multiset contains all words of the length n .

The injective filter $I_{\mathbb{B}}$ consists of those block words which block multiset is a set, i.e. each block appears at most once in a word from $I_{\mathbb{B}}$.

It is clear from the definitions that $I_{\mathbb{B}} \cap S_{\mathbb{B}} = P_{\mathbb{B}}$.

The problems of $I_{\mathbb{B}}$ -realizability and $S_{\mathbb{B}}$ -realizability are decidable. Let’s outline the proof. It turns out that both $I_{\mathbb{B}}$ - or, respectively, $S_{\mathbb{B}}$ - realizability can be reduced to some restricted versions of the integer cone hitting problem whose decidability follows from specific properties of maps Φ defined in Subsection 5.1.

We use the componentwise partial order on the integer orthant \mathbb{Z}_+^d

$$x \prec y \Leftrightarrow x_i \leq y_i \text{ for all } i. \quad (17)$$

Up-hitting Problem

INPUT: a square matrix Φ of order d ; a d -dimensional vector x_0 ; a family of vectors $v_i \in \mathbb{Z}^d, i = 0, 1, \dots, r$.

OUTPUT: ‘yes’ if the *orbit up-shadow* intersects the translate of the integer cone $v_0 + \mathbb{N}(v_1, \dots, v_r) = v_0 + W$ and ‘no’ otherwise. It means that

$$\Phi^n x_0 \prec y \quad (18)$$

holds for some integer n and $y \in v_0 + W$.

The **Down-hitting problem** is defined similarly except for the condition (18) which is replaced by the condition

$$y \prec \Phi^n x_0. \quad (19)$$

Lemma 9. *The $I_{\mathbb{B}}$ -realizability problem is Turing reducible to the down-hitting problem.*

Proof. Repeat the arguments from the proof of Lemma 5. Now it is possible that some binary words in a block word are missed. It means that the condition (9) is replaced by

$$w(\tau) \prec \Phi^n x_0. \quad (20)$$

Applying the arguments from Lemma 6 we see that (20) is transformed to (19) for cones appeared in the reduction from Lemma 6. \square

In a similar way we reduce the surjective filter.

Lemma 10. *The $S_{\mathbb{B}}$ -realizability problem is Turing reducible to the up-hitting problem.*

Proof. In a block word taken from the $S_{\mathbb{B}}$ all binary words of the length n appear (possibly, several times). It leads to the condition (18) for cones appeared in the reduction from Lemma 6. \square

Dickson’s lemma claims that there are no infinite antichains in the poset (\mathbb{Z}_+^d, \prec) . So an orbit up-shadow is a finite union of translated copies of the orthant \mathbb{Z}_+^d . In the case of an orbit down-shadow copies of the orthant are replaced by ‘parallelepipedons’ (the Cartesian products of segments). Thus the up-hitting problem as well as the down-hitting problem is reduced to the nonemptiness check for intersections of integer cones (parallelepipedons), which is an integer linear programming problem, provided the representations mentioned above can be constructed efficiently.

To construct the aforementioned representations we use specific properties of asymptotic behavior of the orbit points $\Phi^n x_0$, where Φ is determined by (8).

Recall that an $e(g)$ -component of $\Phi^n x_0$ is expressed as the number $\nu_n(g)$ of walks of the length n from the vertex id to the vertex g in the graph Γ . The vertex set of the graph Γ is $V(\Gamma) = Q^Q$ and the edge set $E(\Gamma)$ consists of pairs in the form $(f, f f_0)$ or $(f, f f_1)$. (See Subsection 5.1 for a detailed exposition.)

Note that the integer $\nu_n(g)$ is the number of words of the length n in a regular language. This language is recognized by an automaton with the transition graph Γ . Thus the generating function

$$\varphi_g(t) = \sum_{n=0}^{\infty} \nu_n(g) t^n$$

for the language is a rational function and its representation in the form $P(t)/Q(t)$ can be found efficiently.

In the arguments below we need some properties specific to generating functions of regular languages. So it is more suitable for our purposes to analyze the asymptotic behavior in combinatorial settings by considering walks on the graph Γ .

Proposition 9. *For any vertex $g \in V(\Gamma)$ the set*

$$P_g = \{n : \nu_n(g) > 0\}$$

is a semilinear set (a finite union of an exceptional finite set and finite collection of arithmetic progressions) and its description can be constructed efficiently.

Proof. Regard the Γ as the transition graph of a nondeterministic automaton over a 1-letter alphabet. Then the proposition follows from Parikh's theorem. \square

Remark 8. Differences of all progressions in Proposition 9 are the cycle lengths in the graph Γ . So they are divisors of the least common multiple of integers from 1 to $|V(\Gamma)|$. In the sequel we denote this common multiple by N .

Take a vertex g on a directed cycle of the length ℓ . The following inequality

$$\nu_{n+\ell}(g) \geq \nu_n(g). \quad (21)$$

holds. Indeed, one can extend any walk of the length n by the cycle.

Now we divide the vertices of the graph Γ into three groups.

- V_1 consists of vertices v such that some directed cycle (possibly, a loop) goes through the vertex v .
- V_2 consists of vertices v such that there is a walk starting at the id, finishing at the v and passing through a vertex from the set V_1 .
- V_3 consists of all other vertices.

Proposition 10. *If $g \in V_3$ then $\nu_n(g) = 0$ for $n > |V(\Gamma)|$.*

Proof. Any walk of the length $n > |V(\Gamma)|$ from id to g must contain repeating vertices. It means that a part of the walk is a directed cycle. So the walk passes a vertex from the set V_1 . \square

Proposition 11. *The inequality*

$$\nu_{n+N}(g) \geq \nu_n(g), \quad (22)$$

where $N = \text{LCM}(1, \dots, |V(\Gamma)|)$, holds for all $g \in V(\Gamma)$ and $n > |V(\Gamma)|$.

Proof. For $g \in V_3$ apply Proposition 10.

For $g \in V_1$ the inequality (22) follows from (21) and Remark 8.

Now take a vertex $g \in V_2$. The set $V_g \subseteq V_1$ consists of vertices $g' \in V_1$ such that g' belongs to a walk from id to g and for all walks of this type all vertices after the g'

along a walk are in the set V_2 . The subgraph Γ_g is induced by the edges of all walks from a vertex in V_g to g .

Let observe the following properties of the Γ_g .

There are no edges to vertices of the set V_g in the graph Γ_g . Indeed, such an edge contradicts definition of the set V_g .

There are no edges outgoing from the vertex g in the graph Γ_g . Otherwise one would detect a directed cycle passing through g .

From these properties we conclude that the graph Γ_g is acyclic. By definition there are no directed cycles passing through vertices in the set V_2 . Other vertices in the Γ_g are in the set V_g . There are no directed cycles passing through these vertices because there are no edges ingoing to them.

Note also that the maximum of the path length from $g' \in V_g$ to g does not exceed $|V(\Gamma)|$.

From all the properties above we get

$$\nu_n(g) = \sum_{\substack{g' \in V_g \\ k \leq |V(\Gamma)|}} p_{g',k} \nu_{n-k}(g'), \quad (23)$$

where $p_{g',k}$ is the number of paths from g' and g in the Γ_g .

Applying the inequality (22) to all terms in the right-hand side of (23) we get the same inequality for the left-hand side, i.e. for the vertex g . \square

Theorem 7. *The $S_{\mathbb{B}}$ -realizability problem is decidable.*

Proof. It follows from Lemma 10 that it is enough to construct an integer cone representation for an orbit up-shadow. Proposition 11 implies that the orbit up-shadow is

$$\bigcup_{i=0}^N (\Phi^i x_0 + \mathbb{N}^m),$$

where m is a dimension, i.e. the cardinality of Q^Q .

So the problem is reduced to the integer linear programming problem. \square

To prove decidability of the injective filter we should determine for each $0 \leq r < N$ unbounded components of $\Phi^{nN+r} x_0$ and the limit values of the bounded components.

All subsequences $t_g^r(n) = \nu_{nN+r}(g)$, where $0 \leq r < N$, are nondecreasing due to Proposition 11.

A subsequence $t_g^r(n)$, where $g \in V_3$ stabilizes for $n > |V(\Gamma)|$ and the limit value for it is 0.

It follows from (23) that a subsequence $t_g^r(n)$, where $g \in V_2$, tends to infinity iff at least one of the subsequences $t_{g'}^{r-k}(n)$ tends to infinity, where $p_{g',k} \neq 0$.

The remaining case $t_g^r(n)$, where $g \in V_1$, is covered by the following proposition.

Proposition 12. *Let $g \in V_1$. Then $\lim_{n \rightarrow \infty} t_g^r(n) = \infty$ iff there exist a directed cycle C passing through g and an edge (g', g'') such that*

- (i) the edge (g', g'') is not included in the cycle C ;
- (ii) the cycle C passes through the vertex g'' ;
- (iii) $\nu_{nN+r-\ell-1}(g') > 0$, where ℓ is the distance from g'' to g along the cycle C .

Note that due to Remark 8 the conditions (iii) are equivalent for all n . It is clear that (i)–(iii) are verified efficiently.

Proof of Proposition 12. ‘If’ part of the proposition follows from

$$\nu_{(n+1)N+r}(g) \geq \nu_{nN+r}(g) + \nu_{(n+1)N+r-\ell-1}(g') > \nu_{nN+r}(g). \quad (24)$$

Prove now ‘only if’. Suppose that $\nu_{nN+r}(g) = T$ for $n > n_0$. For any directed cycle C passing through g and any edge satisfying (i)–(ii) the first inequality in (24) implies that $\nu_{nN+r-\ell-1}(g') = 0$ for $n > n_0$. \square

Theorem 8. *The $I_{\mathbb{B}}$ -realizability problem is decidable.*

Proof. Determine all unbounded components for all subsequences $\Phi^{nN+r}x_0$ and the limit values for bounded components. For this purpose use Proposition 12 and the observations preceding it. Then the down-shadow is the union of the sets

$$\begin{cases} y_g \geq 0, & \text{if } \lim_{n \rightarrow \infty} (\Phi^{nN+r}x_0)_g = \infty, \\ y_g^\infty \geq y_g \geq 0, & \text{if } \lim_{n \rightarrow \infty} (\Phi^{nN+r}x_0)_g = y_g^\infty. \end{cases}$$

over all $0 \leq r < N$. Here y_g are coordinates in the space \mathbb{Q}^{Q^Q} .

So the problem is reduced to the integer linear programming problem. \square

Now we present an undecidable realizability problem that is related to the $P_{\mathbb{B}}$ -realizability problem.

In the construction we use a track product of languages consisting of block words (*block languages*). Here blocks are words over a finite alphabet Σ and a block word consists of blocks of the same length separated by the delimiter $\#$.

Let Σ_1, Σ_2 be finite alphabets. For the alphabet $\Sigma_1 \times \Sigma_2$ there are two natural projections from $(\Sigma_1 \times \Sigma_2)^*$ to Σ_1^* (resp. to Σ_2^*):

$$\begin{aligned} \pi_1 &: (a_1, b_1)(a_2, b_2) \dots (a_n, b_n) \mapsto a_1 a_2 \dots a_n, \\ \pi_2 &: (a_1, b_1)(a_2, b_2) \dots (a_n, b_n) \mapsto b_1 b_2 \dots b_n. \end{aligned} \quad (25)$$

The *track product* of two block languages L_1 (over an alphabet Σ_1) and L_2 (over the alphabet Σ_2) is a block language $L = L_1 \| L_2$ consisting of all block words over the alphabet $\{\#\} \cup \Sigma_1 \times \Sigma_2$ such that the projection π_1 is in the language L_1 and the projection π_2 is in the language L_2 . (For consistency we assume that $\pi_1(\#) = \pi_2(\#) = \#$.)

Denote by Per_Σ the block language consisting of all periodic words over the alphabet Σ (all blocks of a word in Per_Σ are equal) and by P_Σ the block language consisting

of permutation block words over the alphabet Σ (blocks of a word in P_Σ form the set of all words in Σ^n , where n is the block rank).

The Per_Σ -realizability problem is decidable. Actually it is PSPACE-complete [18]. It turns out that the track product of the periodic filter with the permutation one is undecidable.

Theorem 9. *There are alphabets Σ_1, Σ_2 such that the $(\text{Per}_{\Sigma_1} \| P_{\Sigma_2})$ -realizability problem is undecidable.*

A suitable undecidable problem that is reduced to the $(\text{Per}_{\Sigma_1} \| P_{\Sigma_2})$ -realizability problem is the following.

Zero in the Upper Right Corner Problem. (The ZURC problem.) For a given collection of $D \times D$ integer matrices A_1, \dots, A_N check whether the multiplicative semigroup generated by $\{A_i\}$ contains a matrix M such that $M_{1D} = 0$.

In other words the ZURC problem is to check an existence of an integer sequence j_1, \dots, j_ℓ , where $1 \leq j_t \leq N$, such that

$$(A_{j_1} A_{j_2} \dots A_{j_\ell})_{1D} = 0. \quad (26)$$

Theorem 10 (see [1]). *The ZURC problem is undecidable for $N = 2$ and $D = 18$.*

We will reduce the ZURC problem with $N = 2$ and $D = 18$ to the $(\text{Per}_{\Sigma_1} \| P_{\Sigma_2})$ -realizability problem where

$$\Sigma_1 = [1, \dots, N], \quad \Sigma_2 = [1, \dots, D] \times [1, \dots, D] \times \{0, 1\}. \quad (27)$$

The reduction is similar to the reduction in Section 4.

Let A_1, \dots, A_N be an instance of the ZURC problem for $D = 18, N = 2$. Rewrite the matrices in the form

$$A_j = \begin{pmatrix} \varepsilon_{11}^j m_{11}^j & \varepsilon_{12}^j m_{12}^j & \dots & \varepsilon_{1D}^j m_{1D}^j \\ \varepsilon_{21}^j m_{21}^j & \varepsilon_{22}^j m_{22}^j & \dots & \varepsilon_{2D}^j m_{2D}^j \\ \dots & \dots & \dots & \dots \\ \varepsilon_{D1}^j m_{D1}^j & \dots & \varepsilon_{D(D-1)}^j m_{D(D-1)}^j & \varepsilon_{DD}^j m_{DD}^j \end{pmatrix},$$

where $m_{ik}^j > 0$ and $\varepsilon_{ik}^j \in \{\pm 1, 0\}$. Let M be the maximum of m_{ik}^j .

Fix now a sequence $A_{j_1}, A_{j_2}, \dots, A_{j_\ell}$. Matrix elements in the product have the form

$$(A_{j_1} A_{j_2} \dots A_{j_\ell})_{ik} = \sum_{\tau} \varepsilon(\tau) m(\tau), \quad (28)$$

where τ runs over all sequences of pairs $(i_\alpha k_\alpha)$ such that the length of a sequence is ℓ and $i_1 = i, k_\ell = k, i_{\alpha+1} = k_\alpha$,

$$\varepsilon(\tau) = \prod_{\alpha=1}^{\ell} \varepsilon_{i_\alpha k_\alpha}^{j_\alpha}, \quad m(\tau) = \prod_{\alpha=1}^{\ell} m_{i_\alpha k_\alpha}^{j_\alpha}. \quad (29)$$

Using the expansion (28, 29) we define the partition of words of the length $\ell \cdot \lceil \log_2 M \rceil$ over the alphabet $\Sigma_1 \times \Sigma_2$ into three sets $T_{ik}^+(j)$, $T_{ik}^-(j)$ and $T_{ik}^{\text{bad}}(j)$, where $j = j_1, \dots, j_\ell$. (Below we drop out j while the sequence j is fixed.)

It is convenient to represent a word over the alphabet $\Sigma_1 \times \Sigma_2$, where Σ_i are given by (27), by a 4-row table. A table column represents a symbol in the word. The first row bears symbols from Σ_1 while the remaining three columns represent symbols from Σ_2 (they have three components as indicated in (27)).

A word in the set $T_{ik}^+(T_{ik}^-)$ can be divided in ℓ subwords of the length $\lceil \log_2 M \rceil$. The α th subword has the form

$$\begin{pmatrix} j_\alpha & j_\alpha & \dots & j_\alpha \\ i_\alpha & i_\alpha & \dots & i_\alpha \\ k_\alpha & k_\alpha & \dots & k_\alpha \\ \beta_0 & \beta_1 & \dots & \beta_{\lceil \log_2 M \rceil - 1} \end{pmatrix}. \quad (30)$$

in the table representation defined above. The upper three elements in each row are the same in (30). Note that j_α is the α th element of the sequence j . The fourth row is a binary representation of an integer β .

A word in the set T_{ik}^+ should satisfy the following requirements

- (a) $i_1 = i$;
- (b) $k_\ell = k$;
- (c) $i_{\alpha+1} = k_\alpha$;
- (d) $\beta < m_{i_\alpha k_\alpha}^{j_\alpha}$;
- (e) $\varepsilon(\tau) = 1$, where τ is the sequence of pairs (i_α, k_α) and $\varepsilon(\tau)$ is given by (29).

A word in the set T_{ik}^- should satisfy the requirements (a)–(d) and the modified requirement (e') $\varepsilon(\tau) = -1$.

The set T_{ik}^{bad} collects the rest of words.

Proposition 13. *In notation above*

$$(A_{j_1} A_{j_2} \dots A_{j_\ell})_{ik} = |T_{ik}^+| - |T_{ik}^-|.$$

Proof. Restricting words in the set T_{ik}^+ to the upper three rows one obtains all correct sequences τ such that $\varepsilon(\tau) = 1$. The same restriction for the set T_{ik}^- gives the sequences τ such that $\varepsilon(\tau) = -1$.

The multiplicity of a sequence τ in the set T_{ik}^\pm depends on fourth rows of the tables. According to the requirement (d) and the permutation property there are exactly $m_{i_\alpha k_\alpha}^{j_\alpha}$ subwords bearing $(j_\alpha, i_\alpha, k_\alpha)$ in the upper three rows. So the multiplicity of the sequence τ is $m(\tau)$, where $m(\tau)$ is given by (29). \square

It follows from the above construction that the sets $T_{ik}^\$(j)$, where $\$ \in \{+, -, \text{bad}\}$, do not intersect for different j because the sequence j is recovered from the first row in the table representation.

Proposition 14. *For a fixed collection of matrices A_1, \dots, A_N and for each pair i, k the language*

$$T_{ik}^{\text{all}, \$} = \bigcup_j T_{ik}^{\$}(\mathbf{j})$$

is regular for $\$ \in \{+, -, \text{bad}\}$.

Proof. The requirements (a)–(d) are verified by local check on the subwords of the fixed length $\lceil \log_2 M \rceil$.

Computing the sign $\varepsilon(\tau)$ can be done in the cyclic group of two elements.

So the sets $T_{ik}^{\text{all}, \pm}$ are regular. But the class of regular languages is closed under the complement. Thus the set $T_{ik}^{\text{all}, \text{bad}}$ is also regular. \square

Proof of Theorem 9. We repeat the construction from Section 4. The reduction algorithm takes an instance of the ZURC problem ($N = 2, D = 18$) and constructs an automaton C such that the condition (26) holds for some element in the semigroup generated by the input matrices if and only if $L(C) \cap \text{Per}_{\Sigma_1} \| P_{\Sigma_2} \neq \emptyset$.

The automaton expects an input w from the language $\text{Per}_{\Sigma_1} \| P_{\Sigma_2}$ such that block rank is $\ell \cdot \lceil \log_2 M \rceil$ and the word w is a certificate for zero representation (26). A period in the first (periodic) component determines the sequence $\mathbf{j} = j_1, \dots, j_\ell$ such that (26) holds. The automaton expects that each symbol in the sequence \mathbf{j} is repeated $\lceil \log_2 M \rceil$ times. In the second (permutation) component the automaton expects that the blocks from the sets $T_{1D}^+(\mathbf{j})$, $T_{1D}^-(\mathbf{j})$ are paired and are followed by the blocks from the set T_{1D}^{bad} . Note that such a pairing exists iff $|T_{1D}^+| = |T_{1D}^-|$. The structure of this part of the automaton C is similar to the automaton C' described in Section 4 and shown in Fig. 5.

The correctness of the reduction is proved in a way similar to the arguments in Section 4. If the automaton accepts a word w in the language $\text{Per}_{\Sigma_1} \| P_{\Sigma_2}$ then one can extract the sequence \mathbf{j} from the first components of symbols in the word w . The check in the second components guarantees that (26) holds for the sequence \mathbf{j} . We apply here Proposition 13.

In other direction, if (26) holds for a sequence \mathbf{j} then there exists a word in the language $\text{Per}_{\Sigma_1} \| P_{\Sigma_2}$ satisfying the properties expected (and verified) by the automaton C . Thus $L(C) \cap \text{Per}_{\Sigma_1} \| P_{\Sigma_2} \neq \emptyset$. \square

Remark 9. By suitable encoding of the symbols of the alphabets Σ_1 and Σ_2 one can prove that the $(\text{Per}_{\mathbb{B}} \| P_{\mathbb{B}})$ -realizability problem is also undecidable.

Acknowledgments

Authors are grateful to J. Shallit and I. Sparlinski for helpful remarks.

References

- [1] Bell, P., Potapov I. Lowering undecidability bounds for decision questions in matrices. In Developments in Language Theory. Springer Lecture Notes in Computer Science, vol. 4036, 2006. P. 375–385.

- [2] *Berstel J., Reutenauer Ch.* Rational series and their languages. Springer-Verlag, 1988.
- [3] *Blondel V. D., Portier N.* The presence of a zero in an integer linear recurrent sequence is NP-hard to decide. *Linear Algebra and Its Applications*. 2002. Vol. 351–352. P. 91–98.
- [4] *Cook W., Fonlupt J., Schrijver A.* An integer analogue of Carathéodory’s theorem. *Journal of Combinatorial Theory. Series B*. 1986. Vol. 40, no 1. P. 63–70.
- [5] *Eisenbrand F., Shmonin G.* Carathéodory bounds for integer cones. *Operations Research Letters*. 2006. Vol. 34. P. 564–568.
- [6] *Garey M. R., Johnson D. S.* Computers and intractability: a guide to the theory of NP-completeness. San Francisco: Freeman, 1979.
- [7] *Halava V., Harju T., Hirvensalo M., Karhumäki J.* Skolem’s Problem—On the Border Between Decidability and Undecidability. TUCS Tech. Rep. No 683. 2005.
- [8] *Hopcroft J., Motwani R., and Ullman J.* Introduction to automata theory, languages, and computation. Boston: Addison-Wesley Company, 2001.
- [9] *Laohakosol V., Tangsupphathawat P.* Positivity of third order linear recurrence sequences // *Discrete Applied Mathematics*. 2009. Vol. 157. Issue 15. P. 3239–3248.
- [10] *Parikh R. J.* On context-free languages. 1966. *Journal of the ACM*. Vol. 13, no 4. P. 570–581.
- [11] *Salomaa A., Soittola M.* Automata-theoretic aspects of formal power series. Springer-Verlag, 1978.
- [12] *Schrijver A.* Theory of linear and integer programming. Chichester: John Wiley & Sons, 1986.
- [13] *Shen A., Vereshchagin N. K.* Computable functions. AMS, Providence, RI, 2003.
- [14] *Stanley R.* Enumerative combinatorics. Vol. 1. Monterey: Wadsworth & Brooks/Cole, 1986.
- [15] *Tao T.* Open question: effective Skolem-Mahler-Lech theorem.
<http://terrytao.wordpress.com/2007/05/25/open-question-effective-skolem-mahler-lech-theorem/>
- [16] *Vyalyi M., Tarasov S.* Orbits of linear maps and regular languages. *Discrete analysis and operations research*. 2010. Vol. 17, No 6. P. 20–49 (in Russian).
- [17] *Vereshchagin N. K.* On occurrence of zero in a linear recurrent sequence // *Math. notes*. 1985. v. 38, No 2. p. 177–189.
- [18] *Vyalyi M.N.* On models of a nondeterministic computation. *Proc. of CSR 2009. Springer Lecture Notes in Computer Science*, vol. 5675, 2009. P. 334–345.